V MARATONA DE PROGRAMAÇÃO INTERIF 2022

PRIMEIRA FASE

Caderno de Problemas

Informações Gerais

A) Sobre a entrada

1) A entrada de seu programa deve ser lida da entrada padrão.
2) A entrada é composta de um ou mais casos de teste, depende do problema.
3) Quando uma linha da entrada contém vários valores, estes são separados por um único espaço em branco; a entrada não contém nenhum outro espaço em branco.
4) Cada linha, incluindo a última, contém o caractere final-de-linha.
5) O final da entrada pode coincidir com o final do arquivo ou com uma entrada determinada

B) Sobre a saída

1) A saída de seu programa deve ser escrita na saída padrão.
2) Espaços em branco só devem ser colocados quando solicitado.
3) Cada linha, incluindo a última, deve conter o caractere final-de-linha.

C) Regras

1) Só é permitida a comunicação entre os membros de um mesmo grupo.
2) Não é permitida a comunicação com o técnico (coach) do time.
3) Eventuais dúvidas sobre a prova utilizar o menu “clarification” do sistema de submissão.

D) Ambiente computacional

O sistema de correção das submissões será executado utilizando a distribuição Ubuntu GNU/Linux 20.04.2 LTS amd64, tendo os seguintes compiladores/interpretadores configurados:

- C - gcc version 9.3.0 (Ubuntu 9.3.0-17ubuntu1~20.04)
- C++ - gcc version 9.3.0 (Ubuntu 9.3.0-17ubuntu1~20.04)
- Python 3 - Python 3.8.10
- Java - openjdk-11.0.11
- C# - mono JIT 6.12

Problema A
IFSP Motorsport Championship

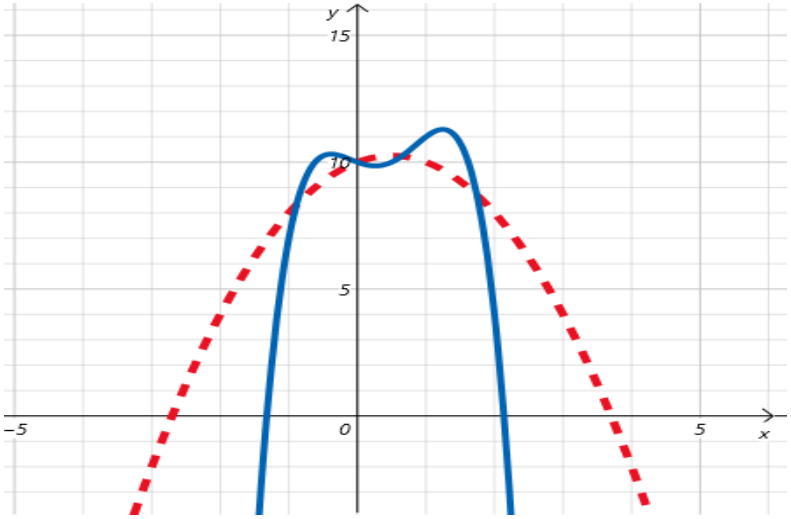
Por Tiago Alexandre Docusse (IFSP – campus Barretos)

Arquivo: motorsport.[c|cpp|java\cs\py]

***Timelimit:** 1*

Hoje em dia existem diversos simuladores virtuais de corridas automobilísticas, e Rafael adora esses simuladores. Recentemente, ele se inscreveu para participar do IFSP Motorsport Championship, um campeonato em que, além de ser um bom piloto, é necessário ser um ótimo engenheiro de motores, já que é possível personalizar alguns aspectos dos motores virtuais antes das corridas. Os motores a combustível de carros podem ser definidos por diversas características, como potência, consumo, torque, entre outros. No caso do IFSP Motorsport Championship, a categoria de carros utilizada no campeonato deste ano é a GT3, que possui carros com diferentes motores e coeficientes aerodinâmicos. Rafael escolheu um carro muito bom aerodinamicamente, porém, com um motor um tanto quanto fraco para essa categoria. Sabendo disso, ele pensou em fazer uma melhoria na curva de torque do seu carro, para ter maior aceleração nas saídas de curva. Rafael estudou a documentação do simulador e descobriu que a curva de torque do carro que ele escolheu foi modelada utilizando um polinômio de segundo grau, que, uma vez atingido o torque máximo, ele decai e é necessário trocar de marcha para que o carro possa ganhar mais torque. Pensando nisso, e sabendo que pode alterar o comportamento do motor do seu carro virtual, Rafael projetou o seu novo motor para que, ao invés de um polinômio de segundo grau, a sua curva de torque seja simulada por um polinômio de quarto grau. Para fazer isso, ele implementou um algoritmo que detecta o torque máximo e aplica um boost temporário no motor, na tentativa de aumentar o torque e conseguir mais velocidade com isso, antes que o torque volte a cair novamente. Dessa forma, a curva de torque do motor virtual deve se comportar, após essa modificação, como um polinômio de quarto grau. Na Figura 1 é exibido um exemplo da curva de torque do motor para uma única marcha, utilizando um polinômio de segundo grau (linha vermelha tracejada) e utilizando um polinômio de quarto grau (linha azul contínua).

Figura 1 – Exemplos de curva de torque



Fonte: criação própria

Agora que Rafael já definiu o que deseja alterar no motor do seu carro virtual, ele pediu sua ajuda. Ao invés de fazer a alteração e testar o carro, ele quer antes saber se o resultado dará certo, analisando os valores da rotação do motor após a utilização do boost, uma vez que tal alteração é complexa e há pouco tempo para o início do campeonato. E aí, você consegue ajudar Rafael a ser o número 1 das pistas virtuais?

Entrada

A entrada do programa deve ser composta inicialmente por uma linha contendo cinco números reais, correspondente aos valores a, b, c, d, e de um polinômio do quarto grau do tipo $ax^4 + bx^3 + cx^2 + dx + e$, tais que $-10^9 \leq a, b, c, d, e \leq 10^9$. Em seguida, informa-se na próxima linha o valor inicial (v_i) e o valor final (v_f) de rotação do intervalo fechado que deseja ser analisado, tais que $-2^{31} \leq v_i, v_f \leq 2^{31} - 1$. Estes valores são números inteiros e podem ser negativos, uma vez que o simulador utiliza valores fictícios para fazer tais cálculos.

Saída

A saída do programa possui duas linhas, a primeira informando o valor (aproximado com sete casas decimais) da rotação no momento em que o torque passa de uma tendência de queda para uma alta, devido à aplicação do boost, e a segunda o valor (aproximado com sete casas decimais) da rotação no momento em que o efeito do boost acaba e o torque passa de uma tendência de alta para uma tendência de queda. Caso o boost não produza a retomada de torque desejada, a saída deve ser formada por uma única linha formada pelas palavras "PARAMETROS FALHOS".

Exemplos de Entradas	Exemplos de Saídas
-2 3 1 0 10 -1 1	-0.0982424 0.8482424
1 -5 0 1 -10 -1 1	PARAMETROS FALHOS

Problema B
Expedição Endurance

Por Rafael da Silva Muniz (IFSP – campus Bragança Paulista)

Arquivo: navio.[c|cpp|java\cs\py]

***Timelimit:** 1*

Fonte: Google Imagens

Em dezembro de 1914 um dos maiores exploradores da Antártica, Ernest Shackleton, iniciava sua segunda expedição ao Polo Sul a bordo de seu navio, o *Endurance*. Porém, no início de 1915 depois de uma forte ventania o *Endurance* foi arrastado ao mar congelante de Weddel. Após ficar mais de 10 meses parado entre dois blocos de gelo, o *Endurance* naufragou. Shackleton tinha que decidir o que fazer o mais rápido possível já que seus suprimentos estavam acabando e ele precisava alimentar diariamente 27 marinheiros e 68 cachorros. A primeira ordem de Shackleton foi de construir uma cozinha e uma cabana com as madeiras que os marinheiros conseguiram resgatar do navio. A segunda ordem foi passada ao cozinheiro para racionar as refeições e o consumo de água. Mesmo estando no meio do gelo, a água doce era restrita e o processo de fusão (transforar gelo em água) consumia muito combustível.

A estratégia criada pelo cozinheiro, para racionar o consumo da água, foi o de lavar a louça somente com o resto da água não consumida na refeição. A ordem da lavagem da louça foi definida pelo cozinheiro como sendo inversa a ordem de entrega na cozinha. Ou seja, a última louça entregue será a primeira a ser lavada, a penúltima entregue será a segunda a ser lavado, a antepenúltima entregue será a terceira a ser lavada. Essa ordem de lavagem deverá ser seguida até a primeira louça entregue que será, respectivamente, a última a ser lavada.

Os marinheiros conseguiram resgatar, antes do naufrágio, 30 itens de cozinha (entre pratos, talheres e canecas) que podem ser usados nas refeições.

Você deve criar um programa para apresentar se todas as louças de uma refeição foram lavadas ou não.

Entrada

A entrada é constituída por N linhas. Cada linha pode apresentar os seguintes valores: 1 (indicando que um prato foi entregue ao cozinheiro), 2 (indicando que um talher foi entregue ao cozinheiro), 3 (indicando que um copo foi entregue ao cozinheiro), 0 (indicando que o primeiro item na ordem da lavagem foi finalizado) e -1 (indicando que a quantidade de água disponível acabou).

Saída

Seu programa deve apresentar na saída **-1** caso o cozinheiro tenha lavado toda a louça de uma determinada refeição ou N linhas indicando as louças que não foram lavadas naquela refeição.

Exemplos de Entradas	Exemplos de Saídas
3 2 2 0 3 1 0 0 1 -1	3 2 1
1 1 2 3 0 0 3 0 2 0 0 2 2 3 1 2 2 0 -1	1 2 2 3 1 2
1 3 0 3 0 0 2 2 1 0 0 0 -1	-1

Problema C
A justiça injusta de dona Florinda

Por Márcio Kassouf Crocomo (IFSP – campus Piracicaba)

Arquivo: main.[c|cpp|java\cs\py]

***Timelimit:** 1*

Dona Florinda é uma senhora muito caridosa, mas que não gosta de injustiças. Sempre em uma dada época do ano, crianças batem na porta de sua casa pedindo por chocolate. Para isso, sempre se prepara comprando 3 caixas de chocolate para esse dia, mas nunca sabe quantas crianças vão aparecer, descobrindo isso apenas no momento em que atende a porta. Cada caixa de chocolate que dona Florinda possui contém um determinado número de chocolates. O objetivo de Florinda é dar o maior número de chocolates possível para que as crianças dividam igualmente entre si, sem que sobre chocolate após essa divisão. Florinda também entrega as caixas fechadas (isto é, não muda a quantidade de chocolate existentes nas caixas compradas). Caso não exista uma combinação de caixas que satisfaça seu critério, nenhuma caixa é entregue, e as crianças ficam sem chocolate. Faça um programa de computador que ajude dona Florinda a encontrar a combinação de caixas que deve ser entregue para as crianças.

Entrada

A primeira linha da entrada possui um número inteiro que pode variar de 1 a 100, representando o número de crianças.

A segunda linha possui 3 números inteiros distintos representando a quantidade de chocolates existentes nas caixas A, B e C, nesta ordem. Cada caixa possui entre 1 e 200 chocolates.

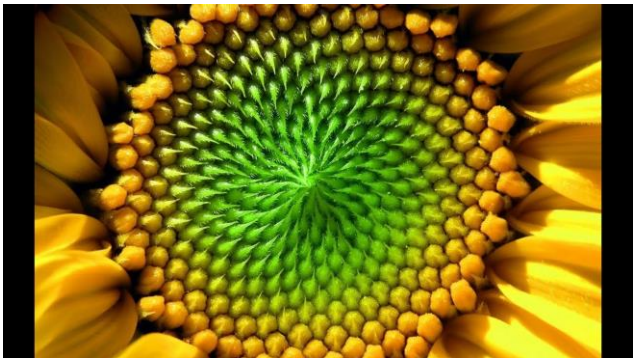
Saída

A saída é composta de 2 linhas. A primeira com as letras das caixas que deverão ser entregues para as crianças, todas maiúsculas e em ordem alfabética. A segunda linha possui um número inteiro indicando a quantidade total de chocolates que será entregue. Caso não exista uma configuração de caixas que satisfaça o critério estabelecido, a primeira linha deverá exibir "NENHUMA" e a segunda linha o número "0".

Exemplos de Entradas	Exemplos de Saídas
5 10 20 30	ABC 60
7 15 3 6	AC 21
15 14 20 88	NENHUMA 0
9 18 2 42	A 18

Problema D
Fibonacci

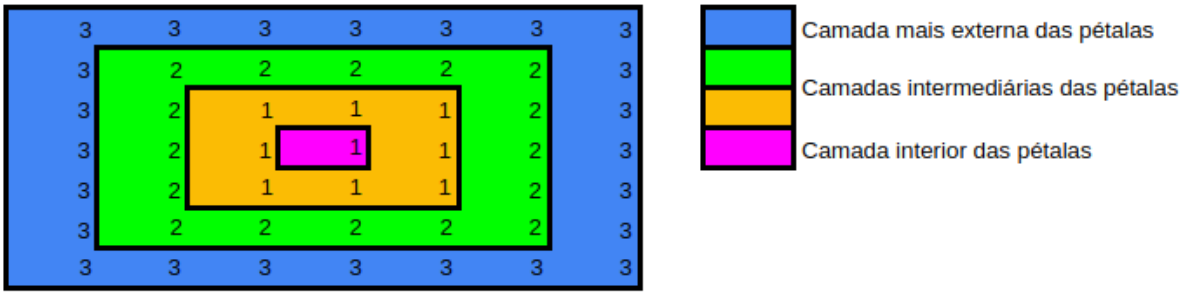
Por Joice Mendes (IFSP – campus Campinas)
Arquivo: fibonacci.[c|cpp|java\cs\py]
***Timelimit:** 1*



A sequência de Fibonacci pode ser encontrada na natureza. É possível encontrar essa série expressa em arranjos de plantas, copas de árvores ou até mesmo nas pétalas das flores ou em alguns legumes. Além disso, o crescimento populacional de alguns animais segue também essa progressão.

Essa sequência, proposta pelo matemático Leonardo Pisa, tem como primeiro e segundo termos o valor 1. Os demais termos, a partir do terceiro, são calculados como a soma dos dois anteriores. Ou seja, F(n) = F(n-1) + F(n-2).

O biólogo Jean Python quer analisar algumas flores, considerando o tamanho de suas pétalas. Ele deseja verificar se aquela espécie se comporta conforme a série de Fibonacci. Para isso, ele representou em uma matriz quadrada os tamanhos das pétalas, de forma concêntrica. Ou seja, a medição iniciou com as pétalas mais interiores (centro da matriz) e, de forma concêntrica, a matriz foi sendo preenchida. Verifique o exemplo abaixo.



Sua tarefa é desenvolver um programa que, após a leitura da matriz de medição das pétalas, informe se a flor segue ou não a sequência de Fibonacci.

Entrada

A entrada é composta por N linhas com valores inteiros. A primeira linha contém a ordem da matriz quadrada. As linhas seguintes apresentam as medições realizadas.

Saída

A saída deverá apresentar a mensagem ‘Fibonacci’ ou ‘Nao fibonacci’.

Exemplos de Entradas	Exemplos de Saídas
3 1 1 1 1 1 1 1 1 1	Fibonacci
3 1 7 2 1 1 1 1 1 1	Nao fibonacci
5 2 2 2 2 2 2 1 1 1 2 2 1 1 1 2 2 1 1 1 2 2 2 2 2 2	Fibonacci
5 2 2 2 2 2 2 1 1 1 2 2 1 7 1 2 2 1 1 1 2 2 2 2 2 2	Nao fibonacci

Problema E
Figurinhas da Copa

Por Jorge Francisco Cutigi (IFSP – campus São Carlos)

Arquivo: figurinhas.[c|cpp|java\cs\py]

***Timelimit:** 1*

Estamos em ano de Copa do Mundo. Mais do que futebol, jogos e gols, um personagem chama muita a atenção do público de todas as idades: O Álbum de Figurinhas da Copa.

Otto gosta muito de futebol e de colecionar figurinhas. Por isso, já adquiriu o álbum e muitas figurinhas. No álbum há N espaços para colar figurinhas de jogadores, brasões, estádios, entre outros. Cada espaço possui um número, o qual identifica unicamente a figurinha (os espaços numerados no álbum sequencialmente de 1 a N). Além das figurinhas tradicionais, existem figurinhas especiais feitas com papel brilhante. Todas as figurinhas com final 3 são especiais (ex: figurinha número 73)

Otto está aprendendo a programar e decidiu criar um programa que, dado o número de espaços para colar figurinhas no álbum e a sequência de figurinhas que ele possui, o programa informa: quantas figurinhas ele colou no álbum; quantas figurinhas especiais ele colou; quantas figurinhas repetidas ele possui; quantas figurinhas especiais repetidas ele possui.

Entrada

A entrada consiste de duas linhas: a primeira contém um número inteiro N que representa o número de espaços no álbum. A segunda linha consiste de uma sequência de números inteiros que representam as figurinhas que Otto possui.

Saída

Seu programa deve imprimir quatro números (X, Y, Z, W) em sequência, onde: X é a quantidade figurinhas que Otto colou no álbum; Y é a quantidade figurinhas especiais ele colou; Z é a quantidade figurinhas repetidas ele possui; e W é a quantidade figurinhas especiais repetidas ele possui. Se alguma das figurinhas informadas for invalida (número maior que o número de espaços do álbum, a saída deve ser a string ERRO.

Exemplos de Entradas	Exemplos de Saídas
25 23 2 12 13 8 2 12 21 21 23 19 12 2	7 2 6 1
10 1 3 6 15 8 3	ERRO

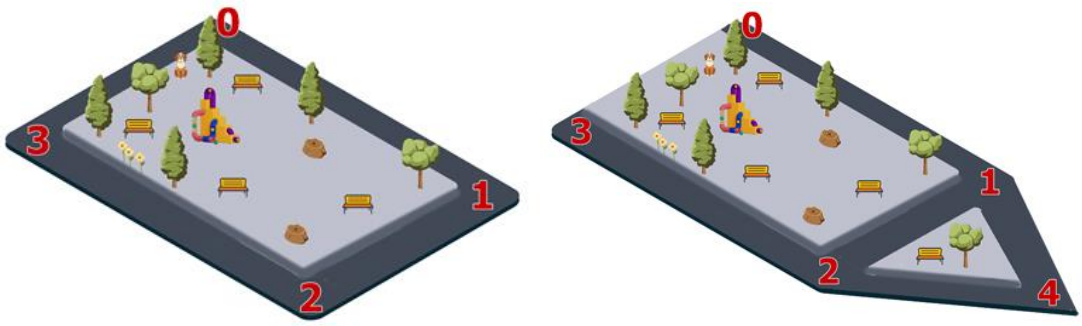
Problema F
Passeio no parque

Por Jones Mendonça de Souza (IFSP – campus Barretos)

Arquivo: passeio.[c|cpp|java\cs\py]

***Timelimit:** 1*

Regina passeia de carrinho com seu bebê todo o final de semana em um dos parques da cidade de Bebelândia. Ela percebeu que seu bebê fica irritado durante o passeio quando passam pelo mesmo trajeto mais de uma vez. Dessa maneira, Regina teve a seguinte ideia, pegar os mapas de todos os parques da cidade e analisar se ela consegue percorrer todos os pontos do parque sem ter que repetir o mesmo trajeto. Por exemplo, analisando o mapa ilustrado na Figura (A), podemos concluir que a partir de qualquer ponto do parque conseguimos passear por todos os pontos sem repetir um mesmo trajeto. Já analisando o mapa ilustrado na Figura (B), podemos concluir que para passear por todos os pontos do parque precisamos repetir um dos trajetos. Isso ficaria mais fácil desenvolvendo um algoritmo, em que dado o mapa de entrada do parque o algoritmo respondesse para Regina se ela consegue passear ou não com seu bebê. Você consegue ajudá-la?



(A) (B)

Entrada

A entrada é composta por um único mapa. Na primeira linha deverá ser informado, um inteiro **N** que indica o número de ligações entre pares de pontos do parque. Depois, separados por um espaço, seguem-se **K** linhas contendo as ligações entre os pontos do parque, representada por dois valores inteiros **O** e **D**.

Saída

A saída deverá apresentar em uma única linha a palavra “Sim”, caso o mapa do parque permita um passeio sem repetir um trajeto, e “Nao” caso contrário.

Exemplos de Entradas	Exemplos de Saídas
4 0 1 1 2 2 3 3 0	Sim
5 0 1	Nao

1 4 1 2 2 3 4 2	

Problema G
Teclado

Por Murilo Varges da Silva (IFSP – campus Birigui)
Arquivo: teclado.[c|cpp|java\cs\py]
***Timelimit:** 4*

Quantas teclas são necessárias pressionar para escrever uma mensagem de texto? Você pode pensar que é igual ao número de caracteres no texto, mas isso só é correto se uma tecla gera um caractere. Com dispositivos de bolso, as possibilidades para a digitação de texto são muitas vezes limitadas. Alguns dispositivos fornecem apenas alguns botões, um número significativamente menor do que o número de letras no alfabeto. Para esses dispositivos, o usuário precisa apertar vários botões para digitar um único caractere. Um mecanismo para lidar com essas limitações é um teclado virtual exibido em uma tela, com um cursor que pode ser movido para selecionar caracteres. Quatro botões de seta controlam o movimento do cursor, e quando o cursor é posicionado sobre uma tecla apropriada, pressionando o quinto botão seleciona o carácter correspondente e anexa-o ao fim do texto. Para finalizar o texto, o usuário deve navegar e selecionar a tecla "Enter". Isso fornece aos usuários um conjunto arbitrário de caracteres e que lhes permite digitar texto de qualquer tamanho, com apenas cinco botões de hardware.

Neste problema, você terá um layout de teclado virtual e sua tarefa é determinar o número mínimo de traços necessários para escrever um texto, onde pressiona qualquer um dos cindo botões de hardware conta como uma tecla pressionada. As teclas estão dispostas numa tabela retangular, de tal modo que cada tecla virtual ocupa uma ou mais unidades da tabela. O cursor começa no canto superior esquerdo do teclado e move-se em quatro pontos cardeais, de tal forma que sempre avança para a próxima unidade quadrada em que a direção que pertence a uma tecla diferente. Se não existe tal quadrado de unidade, o cursor não se move.

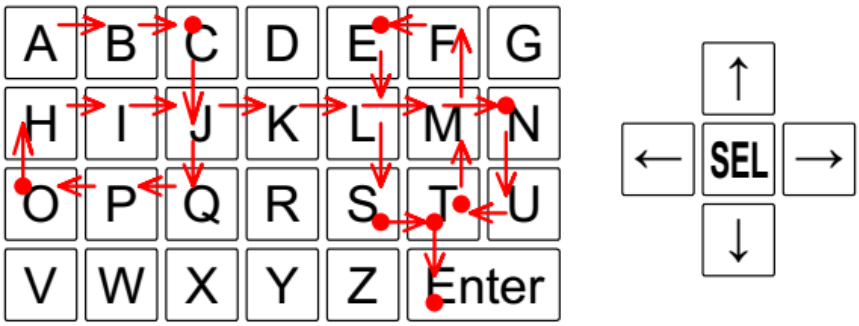


Figura 1 - Exemplo Entrada 1. Um exemplo de um teclado virtual e dos botões em hardware.

Figura 1, ilustra exemplo Entrada 1, mostra uma maneira possível de digitar a palavra "CONTEST" 30 toques em um teclado virtual como o da Figura 1. Os pontos vermelhos representam as teclas virtuais, onde o botão de seleção foi pressionado.

Entrada

A primeira linha da entrada contém dois números inteiros r e c ($1 \leq r, c \leq 50$), dando o número de linhas e colunas da grade do teclado virtual. O teclado virtual é especificado nas próximas r linhas, cada uma das quais contém caracteres c. Os possíveis valores desses caracteres são letras maiúsculas, dígitos, um traço e um asterisco (representando Enter). Há apenas uma tecla que corresponde a qualquer caractere dado. Cada tecla é constituída por um ou mais quadrados de tabela que representa o teclado, que irão sempre formar uma região ligada. A última linha da entrada contém o texto a ser digitado. Este texto é uma string não-vazia de, no máximo, 10.000 caracteres, sem asterisco.

Saída

Exibir o número mínimo de toques necessários necessárias para digitar o texto todo, incluindo a tecla "Enter" no final. Garantido que o texto pode ser digitado.

Exemplos de Entradas	Exemplos de Saídas
4 7 ABCDEFG HIJKLMN OPQRSTU VWXYZ** CONTEST	30
5 20 12233445566778899000 QQWWEERRTTYYUUIIOOPP -AASSDDFFGGHHJJKKLL* --ZZXXCCVVBBNNMM--** ---------- ACM-ICPC-WORLD-FINALS-2015	160
2 19 ABCDEFGHIJKLMNOPQZY X*****************Y AZAZ	19
6 4 AXYB BBBB KLMB OPQB DEFB GHI* AB	7

Problema H
Aquário

Por Felipe Gobo Bruno (IFSP – campus Boituva)
Arquivo: aquario.[c|cpp|java\cs\py]
***Timelimit:** 1*

Diogo é um grande amante de aquários e participa de diversos fóruns sobre o assunto. Analisando essa comunidade, ele sentiu a necessidade de uma solução computacional que ajudasse os iniciantes no hobby, e após aplicar um questionário descobriu que a maior dificuldade entre os novatos é o cálculo da quantidade de litros aquário, a quantidade de substrato que deve ser utilizada e a quantidade de ml do produto para tratar a água.

Pesquisando sobre as dificuldades apresentadas pela comunidade, Diogo descobriu algumas regrinhas para esses cálculos:

- Para descobrir a quantidade de litros, devemos multiplicar as dimensões do aquário e dividir por 1000.

- A quantidade de substrato depende da altura desejada, então substituímos a altura do aquário pelo valor desejado e aplicamos o mesmo calculo anterior e descobriremos a quantidade em quilos.

- A indicação do fabricante para a utilização do produto que trata a água é de 3 gotas por litro, sendo que cada gota representa 0,05ml.

Ajude Diogo a desenvolver essa solução computacional.

Entrada

A primeira linha da entrada possui três números inteiros **N** (1<=N<=2.147.483.647), **O** (1<=O<=2.147.483.647) e **P** (1<=P<=2.147.483.647), representando o comprimento (em centímetros), altura (em centímetros) e largura (em centímetros) do aquário.

A segunda linha da entrada possui um número inteiro **S** (1<=S<=2.147.483.647), representando a altura (em centímetros) desejada para o substrato.

Saída

A saída deve exibir:

Na primeira linha um número ponto flutuante com precisão de 1 (uma) casa decimal A (0<=A<=1038), representando a quantidade de litros do aquário, seguido das letras “LTS”.

Na segunda linha um número ponto flutuante com precisão de 1 (uma) casa decimal B (0<=B<=1038), representando a quantidade de quilos de substrato, seguido das letras “KG”.

Na terceira linha um número ponto flutuante com precisão de 1 (uma) casa decimal C (0<=C<=1038), representando a quantidade de ml do produto para tratar a água, seguido das letras “ML”.

Exemplos de Entradas	Exemplos de Saídas
60 30 40 5	72.0LTS 9.0KG 10.8ML
120 40 70 7	336.0LTS 33.6KG

	50.4ML

40 10 20 0	8.0LTS 0.0KG 1.2ML